Implementación de un gestor de datos y permisos para el desarrollo de aplicaciones de la Universitat Autònoma de Barcelona

Montesinos Santos, Daniel

Resumen—Con el objetivo de dar soporte a una aplicación para la difusión entre los miembros de la comunidad universitaria de los resultados de las encuestas que programa de forma periódica la UAB, se ha creado una herramienta que permite almacenar y gestionar los datos necesarios para implementar un sistema complejo y completo de control de acceso a la información en función del ámbito y de los posibles cargos que pueda ocupar el usuario. Para ello se ha desarrollado una aplicación web para la importación y el mantenimiento de los datos, así como una API que permite conectar cualquier nueva aplicación que requiera un control de permisos que tenga en cuenta los distintos ámbitos a los que puede pertenecer el usuario que accede con su Número de Identificación Universitaria (NIU) y contraseñas institucionales.

Palabras clave— API, Bootstrap, Cargos académicos, Control de acceso, DataTables, Datos universitarios, Encuestas, Gestor de datos, Importación automática, JQuery, JSON, Laravel, OAuth, Postman, REST.

Abstract— In order to support an application for the dissemination among members of the university community of the results of the surveys that the UAB schedules periodically, a tool has been created that allows the storage and management of the data necessary to implement a system complex and complete control of access to information based on the scope and possible positions that the user may occupy. For that purpose, a web application has been developed to import and maintain the data, as well as an API that allows connecting any new application that requires permission control that takes into account the different areas to which the user who accesses may belong. with your University Identification Number (NIU) and institutional passwords.

Index Terms— Academic positions, Access control, API, Automatic Import, Bootstrap, DataTables, Data Manager, JQuery, JSON, Laravel, OAuth, Polls, Postman, REST, University data.

1 Introducción

En este documento se describe el proceso de desarrollo de una herramienta que facilita la gestión de datos de la universidad (centros, departamentos, estudios, profesorado, asignaturas, cargos académicos, …) y la definición de permisos a usuarios en función de los ámbitos a los que pertenecen y los cargos que ocupan.

El proyecto parte de la necesidad de dar soporte a una aplicación para la visualización de los resultados de las encuestas que periódicamente la Universidad pasa a los estudiantes para conocer su nivel de satisfacción sobre las asignaturas que cursan y los docentes que las imparten.

Actualmente la difusión de los resultados de dichas encuestas es muy limitada y por ello se ha propuesto el desarrollo de un proyecto que debe permitir la visualización de resultados en función del tipo de usuario que acceda a la aplicación, definiendo unos permisos a partir del ámbito al que pertenece el usuario y a los posibles cargos académicos que pueda ocupar en un momento determinado. El trabajo que exponemos facilitaría la gestión y definición de permisos, así como el paso de la información a este visor de encuestas.

A lo largo del desarrollo del proyecto se observó que se podía generalizar el uso de la herramienta a otras aplicaciones que tuvieran la necesidad de acceder a datos almacenados en función del tipo de usuario, proporcionando un sistema completo y complejo de gestión de permisos, lo que permite al equipo de desarrollo centrarse en la funcionalidad propia de la aplicación sin la necesidad de disponer de toda la información de control en la propia base de datos.

- E-mail de contacto: Daniel.montesinos@e-campus.uab.cat
- Mención realizada: Tecnologías de la Información
- Trabajo tutorizado por: Jordi Pons Aróztegui (DEIC)
- Curso 2020/21

Por ello se planteó finalmente el proyecto en dos partes: una herramienta para el mantenimiento y gestión de los datos y permisos; y un conector para poder pasar a otras aplicaciones dicha información.

En este informe expondremos los aspectos fundamentales del desarrollo de este proyecto empezando por la descripción de la necesidad de este trabajo, analizando la situación actual de difusión de las encuestas realizadas a los estudiantes. A continuación definiremos los objetivos a conseguir, detallando el nivel de prioridad de cada uno de ellos. Haremos un repaso de la metodología utilizada para el desarrollo, así como de la planificación prevista de tareas. Nos centraremos en los aspectos más importantes del desarrollo de la aplicación y se realizará una descripción detallada de los resultados obtenidos. Por último, para cerrar el documento, se presentarán posibles líneas futuras y las conclusiones extraídas de la realización de este proyecto.

2 Contexto y estado del arte

Como ya hemos comentado previamente, el punto de partida de nuestro proyecto se encuentra en la necesidad de dar una mayor difusión a las encuestas que la UAB realiza anualmente, generalmente a final de semestre, pero no de forma exclusiva. El objetivo de la realización de estas encuestas es comprobar el nivel de satisfacción de los miembros de la comunidad universitaria para introducir acciones de mejora. No solo se realizan a los alumnos, sino que pueden estar dirigidas a profesores y otros estamentos de la comunidad.

Las respuestas a las encuestas permiten comprobar hasta qué punto se han realizado las cosas correctamente y si se pueden mejorar en futuras ocasiones, siendo la opinión de los participantes algo primordial, dado que son los destinatarios de los servicios ofrecidos.

Las encuestas que más predominan en la universidad son las relacionadas con la satisfacción de la impartición de la docencia en las asignaturas y la actuación docente de los profesores. Es por este motivo que cada curso las responden muchos estudiantes.

Pero no es suficiente con recopilar los datos, sino que se debe poder extraer información global para mostrarla a los agentes involucrados, así como poder hacer un análisis completo para facilitar la toma de decisiones para la mejora de la calidad docente.

Actualmente la explotación de estos datos es muy limitada, ya que sólo se entregan informes en un documento *pdf* de resultados a los docentes implicados y a los equipos de dirección de escuela y departamento, impidiendo procesarlos en detalle. También se genera un resumen con datos agregados que se publica en el portal web de la universidad. [1]

Con el objetivo de resolver esta carencia se propone realizar una herramienta que permita a los distintos miembros de la comunidad universitaria poder visualizar los resultados de estas encuestas en función de un conjunto de permisos configurables y establecidos por los órganos competentes. Pero para poder realizar esta visualización se requiere previamente definir la estructura de datos que ha de permitir almacenar toda la información necesaria, facilitar la recuperación de forma automática de los resultados de las encuestas, definir y gestionar los permisos y finalmente establecer el mecanismo de visualización de los resultados.

Debido a la extensión del proyecto, que supera en gran medida el número de horas previsto para un TFG, se dividió en dos proyectos independientes, donde el primero se encargará exclusivamente de gestionar de forma eficiente los permisos, además de mantener los datos y crear una plataforma de comunicación con el segundo proyecto. Este segundo proyecto, será el encargado de gestionar las encuestas y darles visibilidad.

La plataforma que presentamos en este documento será la encargada de ofrecer la información sobre los privilegios y los permisos de los usuarios a la hora de acceder a la herramienta de gestión y visualización de las encuestas, proyecto desarrollado por Sílvia Sanvicente García.

Dada la posibilidad de aprovechar esta infraestructura en otros proyectos, se ha decidido generalizar el ámbito de aplicación y no limitarse a comunicarse únicamente con la aplicación de encuestas, lo que permitirá utilizarla para gestionar cualquier proyecto que almacene elementos que sean compatibles con ésta y que requiera un control de permisos.

3 Objetivos

El objetivo principal del proyecto era facilitar a la herramienta de visión de encuestas un sistema flexible y complejo de gestión de permisos de usuario. Viendo las posibilidades de la herramienta el objetivo pasó a ser el de poder ofrecer este sistema a cualquier aplicación que tenga la necesidad de gestionar permisos de acceso sobre datos de la universidad.

En la TABLA I podemos ver el conjunto de objetivos planteados para poder lograr el objetivo principal. En la tabla establecemos también la prioridad de cada objetivo.

TABLA 1. Resumen de objetivos

Objetivo	**Prioridad**
Establecer un sistema de permisos para usuarios de diferentes niveles jerárquicos, definición de cargos y ámbitos.	Alta
Controlar de forma eficiente los accesos a los datos y su mantenimiento mediante una interfaz gráfica.	Alta

Controlar el gestor de permisos mediante el sistema centralizado de acceso de la universidad (SAC).	Media
Favorecer el tratamiento de volúmenes grandes de datos mediante la importación automática de los mismos.	Alta
Permitir el acceso a la aplicación desde cualquier dispositivo conectado a internet; entorno web y responsivo.	Alta
Garantizar la seguridad de los datos de la base de datos, impidiendo el acceso a usuarios no autorizados.	Media
Establecer un sistema de intercambio de información con el módulo de visualización de encuestas u otras herramientas que requieran un gestor de permisos de acceso.	Alta

Para conseguir estos objetivos se observa la necesidad de definir tres grandes bloques:

- Desarrollo de la base de datos (prioridad alta): Diseño e implementación de una base de datos afín que permita cumplir con todos los requisitos necesarios.

- Desarrollo del gestor (prioridad alta): Creación de una primera aplicación con todas las funcionalidades y opciones necesarias para crear, modificar, eliminar y asignar cualquier elemento de la base de datos.

- Desarrollo del conector (prioridad alta): Creación de una segunda aplicación encargada de satisfacer las peticiones recibidas con la información almacenada.

4 METODOLOGÍA Y PLANIFICACIÓN

4.1 Metodología

Para el desarrollo del proyecto se ha decidido utilizar la metodología en cascada [2].

De esta forma se han definido los requisitos al inicio, se ha creado un diseño afín al propósito del software, se ha implementado, se ha verificado la correcta implementación y cumplimiento de los requisitos y, por último, se ha procedido al mantenimiento de la plataforma.

Por otra parte, la aplicación que se encarga de gestionar los permisos, se ha programado siguiendo una metodología evolutiva, donde se ha podido ir viendo el continuo desarrollo en versiones preliminares.

Para el conector, se planteó un estudio de alternativas y se decidió implementar una API REST [3], analizando el conjunto de consultas posibles requeridas por las aplicaciones a las que se pretende dar servicio.

Para favorecer los tiempos de ejecución y facilidad de uso, todo el desarrollo se ha realizado en local, incorporando continuamente al servidor nuevas versiones y comprobando el correcto funcionamiento de éste.

El servidor donde se han instalado finalmente las herramientas desarrolladas es deic-projectes.uab.cat, equipo proporcionado por el Departament d'Enginyeria de la Informació i les Comunicacions donde se encuentran distintas aplicaciones de soporte a la coordinación de estudios del grado en ingeniería informática.

Mediante Kanban, se han ido organizando las tareas realizadas, pudiendo ver de forma sencilla y organizada las preferencias de desarrollo.

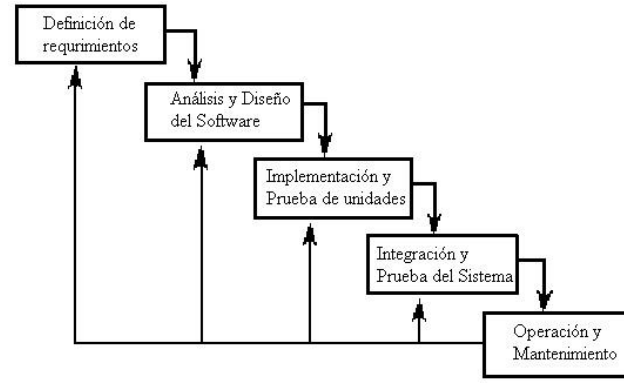

Ilustración 1. Metodología en cascada

4.2 Planificación

La planificación ha sido una tarea importante, teniendo en cuenta que el proyecto tenía grandes dependencias entre los componentes principales.

Para facilitar esta tarea, se ha recurrido al diseño de un diagrama de Gantt que podemos ver en la [Ilustración 16. Planificación inicial del proyecto] del Anexo.

Se ha dividido esencialmente en cinco partes. Una primera parte para la instalación de todas las herramientas y componentes necesarios. Una segunda parte para realizar el diseño del proyecto y comenzar con la implementación del esqueleto básico. Posteriormente, una tercera fase con la finalidad de implementar todas las funcionalidades del gestor. La cuarta fase estaría encargada de implementar la aplicación conectora, a la cual se conectarán las demás aplicaciones. Por último, una fase destinada a cerrar el proyecto y comprobar que los resultados de las fases anteriores cumplen con todos los objetivos y requisitos.

A lo largo del desarrollo del proyecto se ha seguido la planificación prevista sin necesidad de realizar cambios importantes, consiguiendo finalizar las etapas propuestas dentro del calendario propuesto.

La parte final de la planificación pretendía desplegar el proyecto sobre uno de los servidores de la universidad. Por motivos de incompatibilidad entre las versiones de PHP del entorno local de desarrollo y del servidor de la universidad, no ha sido posible llevarlo a cabo.

5 DESARROLLO

5.1 Herramientas y lenguajes de programación

Para conseguir los objetivos propuestos el proyecto se ha desarrollado en un entorno web. Para la implementación del esqueleto y diseño de la página del gestor se ha utilizado HTML5, CSS y PHP, acompañados de JavaScript, JQuery y Ajax.

JQuery y Ajax han facilitado algunas tareas de JavaScript. Por ejemplo, la carga asíncrona de datos por parte del servidor utilizando Ajax o la simplicidad de programar algunas características con JQuery y sus funciones predefinidas de JavaScript.

Inicialmente se decidió realizar todo el proyecto sin ninguna clase de framework, para de esta forma diseñar todo desde cero y a medida. Aun así, más adelante fue necesario incorporar Laravel [4] para implementar la API REST, debido a las restricciones de tiempo sobre el proyecto y los requisitos de seguridad como ya veremos más adelante.

También se ha recurrido al uso de Bootstrap [5] para el diseño del gestor. Esto ha facilitado el desarrollo del entorno responsivo y el diseño gráfico en general.

La incorporación automática de los datos de los ficheros Excel requería de la implementación de una librería externa. Se han barajado múltiples opciones y finalmente se ha escogido PhpSpreadsheet [6] por su simplicidad de uso y características.

Por último, SQL ha sido el lenguaje utilizado para el desarrollo y gestión de la base de datos.

5.2 Entorno

El proyecto se ha realizado en local mediante la herramienta XAMPP [7]. Esta herramienta incluye Apache/2.4.46, PHP/7.4.10 y MariaDB junto a PhpMyAdmin para una fácil gestión de la base de datos.

El diseño inicial de la base de datos se realizó sobre MySQL Workbench [8], dado que es una potente herramienta que facilita la detección de fallos.

Para el desarrollo de la aplicación API y generar consultas de prueba se ha buscado soporte en la herramienta Postman [9].

Por último, para el control de versiones y backups de este código tan extenso, se ha recurrido a la plataforma de GitHub [10] donde se han hecho subidas periódicas.

5.3 Estructura del código

Para el desarrollo del código, se ha seguido el patrón Modelo-Vista-Controlador (MVC) [11].

Gracias a esta estructura se pueden limitar los accesos a los recursos mediante los controladores que, apoyados en los modelos, permiten mostrar unas vistas u otras.

Además, esto ha facilitado la organización de todos los ficheros de forma estandarizada para futuras mejoras y ampliaciones.

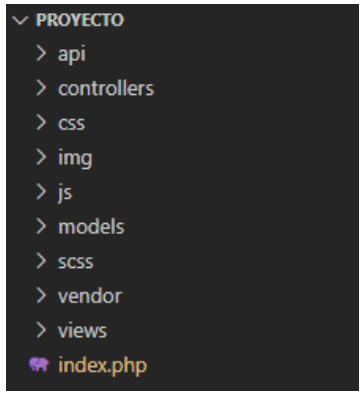

Ilustración 2. Organización de carpetas del proyecto

5.4 Base de datos

La base de datos ha sido un aspecto crítico para el correcto desarrollo del proyecto. Es por ello que se ha recurrido a modificar el diseño en varias ocasiones después de cada reunión con el tutor. Esto ha sido necesario debido a que iban surgiendo nuevas necesidades o se encontraban limitaciones que no se habían previsto inicialmente.

El objetivo de la base de datos es almacenar todos los datos relacionados con el proyecto, empezando por todos los centros y facultades que forman la universidad, los estudios que se imparten en ellos, las asignaturas y grupos que los componen, así como los profesores y departamentos implicados en dicha docencia. Un conjunto importante de datos estrechamente relacionados y con muchas dependencias.

Para poder mostrar el correcto funcionamiento de la herramienta se han utilizado los datos de la *Escola d'Enginyeria* y del Grado en Ingeniería Informática.

Por otro lado, la base de datos también debe almacenar la información necesaria para la gestión de permisos. Para ello se definen distintos ámbitos a los que se pueden asignar permisos. Los distintos usuarios pertenecerán a estos ámbitos en función del estamento, del departamento, de las asignaturas que impartan y los posibles cargos académicos que ocupen. Concretamente se han definido los siguientes ámbitos: Centros, Estudios, Asignaturas, Departamentos, Grupos, Profesores, Universidad y Estudiante.

A su vez se han definido inicialmente tres niveles básicos de permisos: ninguno, básico y avanzado, que se podrán asignar a los distintos objetos a gestionar para cada uno de los ámbitos propuestos. De todas formas, el diseño

de la aplicación permite añadir nuevos niveles directamente desde la base de datos. También se podría modificar el código del gestor para incorporar esta funcionalidad.

Podemos ver el diagrama de la base de datos en la [Ilustración 17. Base de datos] del Anexo.

5.5 Aplicación principal

Antes de empezar con la implementación de funciones en la aplicación principal, se diseñó un esqueleto con Bootstrap con la finalidad de mantener los menús de forma estática durante toda la estancia en la página. De esta forma, sólo se actualiza el contenido del container principal en función de las selecciones del usuario.

El esqueleto diseñado ha sido el siguiente:

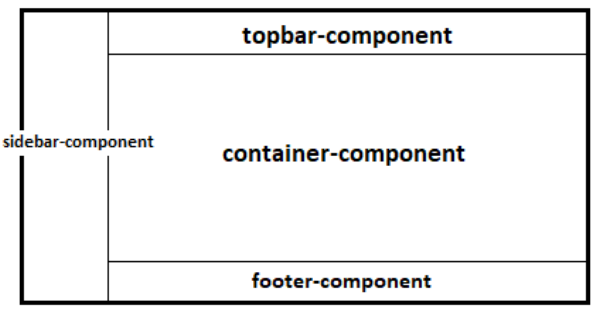

Ilustración 3. Esqueleto del gestor

Para que las herramientas de Bootstrap, JQuery y Ajax funcionen correctamente, se necesita la instalación de sus respectivas librerías. Una solución podría haber sido enlazar el proyecto con el repositorio online de la página oficial, manteniendo siempre actualizadas las librerías. En este caso se ha optado por instalar las librerías de forma local. Asimismo, aunque el repositorio online no esté disponible, se podrá seguir ejecutando el proyecto sin problemas. Además, se eliminan futuras incompatibilidades por actualizaciones inesperadas en cualquier librería.

Un pequeño ejemplo de código con JQuery y Ajax es el siguiente, donde se encuentra una función *load* propia de Ajax encargada de cargar el contenido especificado en el contenedor *container-fluid*.

```
function mostrarMenuCentros() {
    $(document).ready(function(){
        $('#menuCentros').click(function(){
            $('#container-fluid').load('index.php?accion=centros', function () {
                console.log('Carga satisfactoria de centros.');
            });
        });
    });
}
```

Ilustración 4. Función mostrar menú de centros

Además, una de las grandes ventajas de Bootstrap, reside en que proporciona la habilidad a la aplicación de visualizarse correctamente en cualquier tamaño de pantalla. De esta forma, se puede utilizar la aplicación desde cualquier dispositivo indistintamente del tamaño de su pantalla o navegador utilizado para acceder.

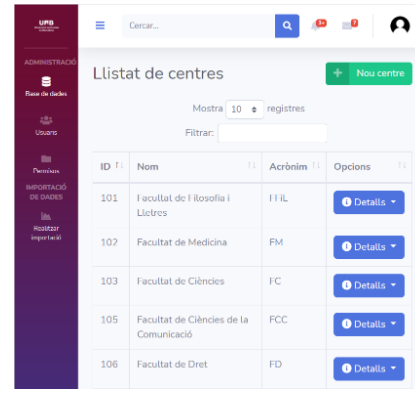

Ilustración 5. Listado de centros desde una pantalla pequeña

Para dar solución a la gestión de los grandes volúmenes de datos de las tablas se ha instalado la librería DataTables [12], la cual ofrece una gestión avanzada de las mismas. De esta forma, se dejan atrás las tablas estáticas comunes y se añaden unas más inteligentes que permiten trabajar con los datos de forma dinámica.

Estas tablas ofrecen paginación, ordenación e incluso filtrado de los resultados.

Ilustración 6. Búsqueda en la tabla objetos

No todos los datos pueden reflejarse mediante tablas. En el caso de la universidad, se encuentran multitud de centros, los cuales incluyen multitud de estudios cada uno y, a su vez, multitud de asignaturas en cada estudio. Si se mostrasen todos estos datos en forma de tabla, se observaría una tabla de dimensiones poco prácticas para que el usuario pudiese trabajar con ella, ni siquiera pudiendo utilizar los filtros de datos. Además, a nivel de servidor también supondría un coste muy elevado la carga constante de todos estos elementos.

Para solventar esto se han implementado listas encadenadas que se generan dinámicamente en función de la selección. Uno de los usos que se le ha dado a esta herramienta, es la de mostrar los grupos de cada asignatura. A continuación, se muestra un ejemplo dónde se ha seleccionado el centro de *Escola d'Enginyeria*, el estudio de *Grau en enginyeria informàtica* y, por último, la asignatura de *Xarxes*.

Ilustración 7. Visualizar los grupos de una asignatura

5.5.1 Gestión de datos

Dado que se deben gestionar los datos, se ha visto la necesidad de agregar también la posibilidad de añadir nuevos elementos manualmente, modificar los existentes y eliminarlos. Aunque se ha implementado un proceso para la importación automática de datos a partir de un documento Excel que se obtiene de las aplicaciones corporativas de la universidad (como veremos en la próxima sección) y, por lo tanto, se presupone que los datos importados son correctos, es posible que sean incompletos en el momento de la importación o que sea necesario modificarlos o, incluso, poder añadir alguno nuevo.

Por tanto, para todos los tipos de datos que se gestionan desde el proyecto se incluyen dichas funcionalidades y se pueden agregar, modificar y eliminar elementos en cualquier momento, siempre y cuando no incurran problemas de dependencias en la base de datos.

La siguiente imagen representaría la creación de un nuevo estudio en la base de datos.

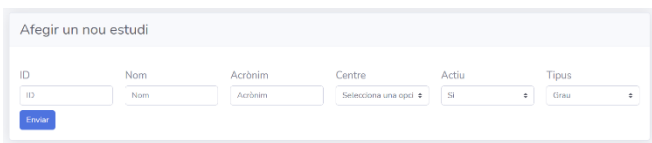

Ilustración 8. Añadir nuevo elemento

Es importante tener en cuenta que las claves primarias en la base de datos no pueden estar repetidas, por lo que si se agrega un elemento cuya clave primaria ya existe, el sistema retornará un error y propondrá intentarlo de nuevo.

En la mayoría de los casos, aunque no siempre, los campos que componen un elemento también pueden ser modificados, a no ser que se trate de la clave primaria del elemento. Modificar esos valores podría dar lugar a inconsistencias en las relaciones y causar futuros problemas en la base de datos.

Un ejemplo de modificación de un estudio podría ser el siguiente:

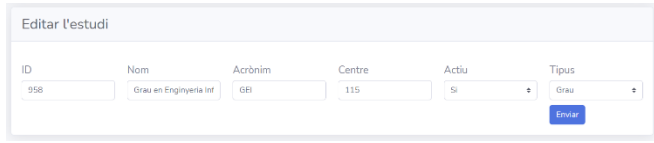

Ilustración 9. Modificar un elemento existente

5.5.2 Importación automática de datos

Como se ha mencionado anteriormente, el proyecto pretende abarcar toda la universidad, lo que incluye numerosos centros, estudios y asignaturas, además de profesores y grupos de asignatura. Esto supone un reto a la hora de incorporar toda esta información anualmente, dado que manualmente sería un trabajo muy costoso y lento.

Desde las aplicaciones corporativas utilizadas en la universidad, se pueden exportar ficheros Excel con los datos que se utilizan en la aplicación y mediante la librería PhpSpreadsheet [6] importarlos en la base de datos de nuestra aplicación con tan sólo una simple acción.

Para ello se ha habilitado una sección donde se debe seleccionar el fichero Excel a importar. Este fichero es leído por un script diseñado a medida que gestiona prácticamente la totalidad de los datos y los incorpora en nuestra base de datos.

Ilustración 10. Importación automática de datos

El script está diseñado específicamente para el fichero Excel que se proporcionó como plantilla, por lo que, si se intenta subir otro fichero distinto o con un formato diferente de tablas, no será posible la importación y retornará error.

Este script es extenso y controla todos los datos de interés del documento Excel, incluyendo la comprobación del año del curso, el estudio y centro al que va dirigido, las asignaturas, profesores y grupos y número de alumnos matriculados en cada grupo.

Esto implica que la ejecución del script se demora unos segundos en función del volumen de datos a importar. Por ejemplo, el documento Excel del curso 2020-21, con más de 280 entradas y más de 20 columnas (más de 5.600 elementos), tarda alrededor de 5 segundos en realizar la importación completa y generar las múltiples dependencias relacionales entre tablas que son necesarias.

5.5.3 Gestión de asignaciones

Una vez la gestión de los datos ha sido completamente incorporada al proyecto, se ha implementado el conjunto de funcionalidades que habilitan a la aplicación para realizar el conjunto de relaciones necesarias entre los datos.

Principalmente se necesitaba poder asignar profesores a sus respectivos departamentos, así como definir cargos académicos y asignarlos a los docentes que los ocupan dentro de la jerarquía de la universidad.

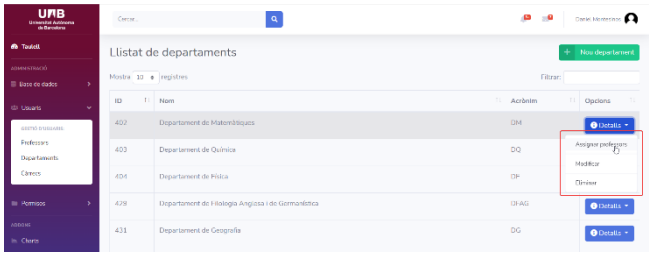

Ilustración 11. Listado de departamentos

Dado que se requería agregar a múltiples profesores a un departamento o incluso, algún cargo podía estar compuesto por más de un profesor, se ha visto la necesidad de utilizar algún sistema de multiselección que agilice esta tarea.

Para ello se ha programado un desplegable que permite filtrar los resultados y seleccionar múltiples opciones de una sola vez. De esta forma, buscando por el nombre, apellido o NIU de un profesor, se le puede localizar rápidamente y agregarlo a la lista de integrantes.

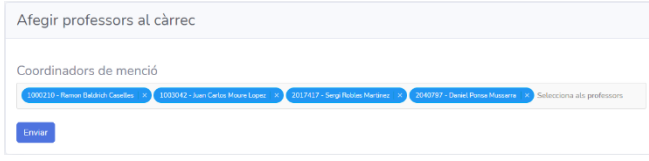

Ilustración 12. Multiselección de profesores a un cargo

5.5.4 Gestión de permisos

Por último se han añadido las funcionalidades necesarias para poder definir el sistema de permisos sobre los objetos definidos en la base de datos. Se ha pretendido desarrollar un procedimiento que permita al usuario final gestionar los permisos de forma sencilla e intuitiva, realizando pocas acciones.

Es por ello que también se ha implementado un sistema de selección de múltiples opciones donde el usuario final puede seleccionar para un objeto concreto, todos los permisos relacionados con todos los ámbitos existentes.

Cabe mencionar que la creación de los nuevos objetos dispara también la creación de sus respectivos permisos por defecto. Se ha creído conveniente que, por defecto, todos los objetos se creen sin permisos para ningún ámbito de aplicación, lo que da un extra de seguridad para el sistema y evita posibles despistes a la hora de asignar los permisos.

Para asignar los permisos a un objeto, se seleccionará éste y se marcará la opción de *asignar permisos*.

Ilustración 13. Listado de objetos almacenados

Una vez marcada la opción, se mostrarán todos los permisos de cada ámbito asociados a este objeto. Es aquí donde se pueden seleccionar los permisos individualizados por ámbito y posteriormente guardarlos.

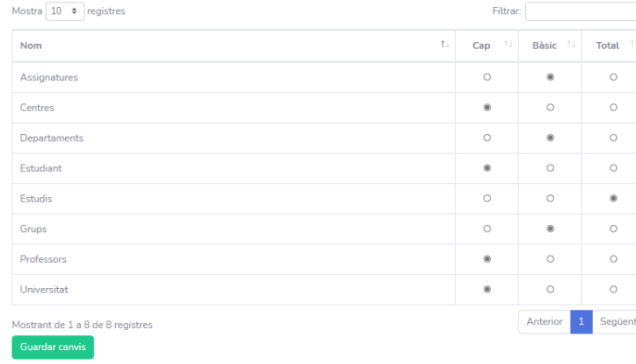

Ilustración 14. Permisos del objeto seleccionado

5.5.5 Seguridad

Se ha prestado especial atención en la seguridad del gestor, es por ello que se han implementado diversas técnicas que favorecen a tener un entorno controlado y seguro.

La primera implementación ha sido la utilización de programación orientada al uso de métodos POST y no GET. En muchas ocasiones la información que viaja entre el cliente y el servidor puede ser comprometida, como información personal de usuarios o acciones de administración contra el servidor. Por este motivo, se ha prescindido del uso del método GET en la mayoría de situaciones, donde dicha información viaja sobre la URL de la página y queda visible para cualquiera.

Todas las características que manipulan información sensible se han programado mediante la metodología POST. Esto es conveniente dado que, combinándolo con protocolos como HTTPS, se puede encriptar todo el tráfico entre el cliente y el servidor, impidiendo la lectura de terceros no deseados.

Otra medida de seguridad importante que se ha implementado es el uso de parametrización en todas las consultas a la base de datos.

Es común encontrar ataques de *SQL-Injection* en los inputs de los usuarios. Es por ello que la parametrización de todos los métodos de entrada del usuario ha sido una prioridad en el desarrollo.

Por último mencionar que se han controlado los inputs para evitar el *Cross-Site Scripting*. Es habitual encontrar ataques donde se pretende ejecutar código desde los inputs del usuario. Es por ello que se deben tratar los datos introducidos antes de realizar cualquier acción. Una forma eficaz es utilizar la función *htmlspecialchars* de PHP.

```
<?php foreach ($lista as $profesor): ?>
    <tr>
        <td><?php echo htmlspecialchars($profesor['niu']);?></td>
        <td><?php echo htmlspecialchars($profesor['nombre']);?></td>
        <td><?php echo htmlspecialchars($profesor['apellido']);?></td>
        <td >
```

Ilustración 15. Ejemplo de control de inputs

Llegados a este punto, hemos visto que las funcionalidades del gestor encargadas de la administración de los datos y permisos quedan resueltas. No obstante, se requiere de un sistema para comunicar y proveer la información de la aplicación gestora con otras aplicaciones interesadas en la información almacenada. En el siguiente punto se explicará en detalle cómo se ha resuelto esta necesidad mediante la aplicación conectora.

5.5.6 Autenticación SAC

Dada la necesidad de controlar el acceso de los usuarios que se conectan a la aplicación gestora, se ha integrado el servicio del SAC (Servei d'Autentificació Centralitzat) [13] de la universidad como herramienta de autenticación de la aplicación. De esta forma añadiendo el NIU del usuario a la tabla Administradores y obligándole a autenticarse, podemos comprobar si este existe en la base de datos y permitirle el acceso a la herramienta. Con ello evitamos la necesidad de guardar contraseñas de la propia aplicación en la base de datos.

5.6 API REST

El siguiente paso reside en la creación de la API, la cual permite a otra aplicación externa solicitar información de nuestra base de datos.

Inicialmente se planteó desarrollar la API completamente desde cero, siendo esta de complejidad baja para cumplir con las necesidades del proyecto. Se desarrolló una primera versión a la cual se le empezaron a ver carencias rápidamente.

La principal carencia residía en la poca seguridad a la hora de solicitarle información. Para el proyecto es necesario filtrar, de alguna forma, las peticiones que se realizan a la API y restringir quien puede realizarlas. Programar esto desde cero suponía un reto que difícilmente podría haber sido cumplido con el tiempo restante. No era la única problemática ya que formatear las URL y crear un espacio de rutas eficiente también suponía un serio problema. Por último y no menos importante, se prevé una demanda de peticiones que, en algunas épocas del año, podría llegar a ser elevada, lo que se traduce en la necesidad de una buena gestión de la cola de peticiones y solicitudes. Dadas estas circunstancias, se creyó conveniente buscar una alternativa adecuada.

La alternativa que más se adaptaba a las necesidades del proyecto ha sido Laravel [4]. Mediante la librería de Laravel se ha podido diseñar la API desde cero, de forma eficiente y con mucho código ya prediseñado para ayudar al desarrollador a integrar funciones complejas de forma sencilla.

El principal requisito de esta API es la necesidad de autenticar los programas que tienen acceso a ella y que no cualquier aplicación pueda realizar peticiones.

Para ello se ha desarrollado un sistema de autenticación mediante una petición POST. Los usuarios envían su identificador y contraseña y si la autenticación es válida, se le devuelve un token que le permite realizar sus peticiones. Concretamente se trata de un token OAuth 2.0 [14].

Todas las pruebas como en la [Ilustración 18. Petición de autenticación con Postman] del Anexo se han realizado con la herramienta Postman que permite realizar diferentes tipos de peticiones a la API y devolver las respuestas de esta.

El token generado y devuelto por la API en la [Ilustración 19. Resultado de la autenticación con la API] del Anexo, se debe utilizar en la cabecera de cada petición que se le haga. El formato de la cabecera debe ser *Authorization* <*tipo*> <*token*>, donde el tipo es Bearer y el token, el que nos haya devuelto la API.

La respuesta del servidor siempre será en formato JSON [15], con una respuesta de error en caso de haberlo o con el contenido de la consulta.

Se han implementado un total de once peticiones posibles a la API. Podemos encontrar una lista en la [Ilustración 22. Lista de peticiones a la API] del Anexo. En esta tabla se pueden ver los tres tipos de peticiones que podemos realizar (GET, POST y DELETE) y el formato de consulta que se debe realizar.

El número de peticiones se puede incrementar fácilmente siguiendo los modelos de las peticiones desarrolladas, en función de las necesidades de las nuevas aplicaciones que puedan requerir su uso.

7 Líneas de futuro

Antes de entrar en detalles de posibles mejoras para el proyecto el principal objetivo debería ser el despliegue en los servidores de la universidad. Al encontrar una incompatibilidad de versiones en el lenguaje de PHP se deberían modificar las librerías utilizadas en el proyecto y adaptarlas, pero esto supondría un gran esfuerzo y posiblemente una limitación en cuanto al uso de la tecnología presente. La solución más idónea reside en la actualización de los servicios de PHP del servidor de la universidad o en la creación de un entorno nuevo que posea las dependencias necesarias para el despliegue del proyecto en su estado actual.

Una vez solucionados los problemas de migración con el servidor final de la universidad el siguiente objetivo debería ser la integración con la herramienta de encuestas, dado que éste era su objetivo desde un principio.

Durante el proyecto se ha programado todo el código pensando en la simplicidad de éste e intentando que sea lo más estándar posible. Esto ofrece la posibilidad de mejorar el trabajo realizado y añadir nuevas características si fuese necesario.

Una característica interesante podría ser la implementación de nuevas opciones para el servicio de importación automática de datos, pudiendo seleccionar que campos se desean importar o mejorando la compatibilidad con otros formatos de fichero y estructura. Además, se podría añadir la funcionalidad inversa a la importación y permitir exportar los datos almacenados en la base de datos del proyecto, con el fin de generar informes de interés.

Otra funcionalidad muy interesante teniendo en cuenta la cantidad de datos que almacena la base de datos del proyecto, sería la de extender la cantidad de funciones que permite responder la API. Actualmente responde las peticiones más comunes y necesarias, sobre todo para el uso específico de la aplicación de encuestas. Esto deja un gran margen para añadir funcionalidades de todo tipo.

Incrementar la cantidad de permisos que se pueden otorgar a cada objeto también sería un gran añadido para el proyecto. Actualmente solo se permiten tres permisos distintos (ninguno, básico o avanzado) dado que son los necesarios para las encuestas, pero se podrían añadir más sin necesidad de realizar muchas modificaciones en el código. De esta forma se podría jerarquizar con mayor precisión los diferentes permisos.

Aunque se han implementado sistemas de autenticación los datos que circulan entre la aplicación y el usuario o la aplicación conectora y la aplicación a la que da soporte, viajan sin ninguna clase de cifrado. Por tanto, un tercero sin autorización podría capturar los paquetes que circulan por la red y leer sin dificultad el contenido que transportan. Un punto prioritario en cuanto a seguridad, sería la implementación de un protocolo de cifrado para el tráfico entre los diferentes nodos que se comunican. Para solventar esta problemática, bastaría con la implementación del protocolo HTTPS. Al utilizar SSL/TLS, se garantizaría que los datos que navegan por la red irían cifrados y aun siendo capturados por un tercero, serían ilegibles.

Durante los requisitos del proyecto se establecieron como posibles cargos, sólo aquellos que pertenecen a cargos académico. Esta es una limitación que podría desvanecerse fácilmente expandiendo el nivel de ámbitos al Personal de Administración y Servicios (PAS) o incluso a cargos concretos de estudiantes (consejo, delegados, …).

Por último una mejora técnica interesante residiría en plantear un paso de la aplicación a una arquitectura de microservicios [16]. Esta mejora no es tan evidente como las demás mencionadas debido a que supondría una modificación bastante más agresiva del proyecto ya realizado. Aun así, la base que se ha logrado diseñar sería un buen punto de partida.

Es un proyecto que queda muy abierto a posibles mejoras y ampliaciones, dado que tiene una sólida base y un amplio abanico de aplicaciones. Mejorando algunos aspectos de los permisos o añadiendo usuarios de otros ámbitos (estudiantes i PAS), se podrían ampliar las posibilidades de conexión con otras aplicaciones (préstamo de material, reserva de espacios, …).

8 Conclusiones

Llegados a este punto es el momento de hacer balance de los distintos aspectos relacionados con el proyecto desarrollado.

En primer lugar es necesario remarcar la evolución que ha tenido lugar desde la propuesta inicial, en la que se pretendía desarrollar una aplicación para la difusión de los resultados de las encuestas de la universidad y que se ha convertido finalmente en un gestor de datos y permisos útil para cualquier aplicación que pueda requerir este servicio.

El trabajo estaba asignado a dos estudiantes y la parte que tenía yo asignada era la gestión y mantenimiento de datos y la definición de permisos de acceso a las encuestas. La otra parte, el gestor y visualizador de encuestas lo ha desarrollado mi compañera Sílvia Sanvicente. Desde un principio trabajamos de forma independiente, definiendo unos objetivos específicos para cada proyecto, evitando al máximo interdependencias y que cualquier retraso en uno de los proyectos no afectase al otro. Los dos trabajos han llegado a su fin con un nivel elevado de consecución de los objetivos, pero no se ha podido realizar una integración completa por falta de tiempo y limitaciones en las versiones del software del servidor.

Las herramientas desarrolladas en el proyecto, aunque mejorables como hemos especificado en el apartado anterior, van a facilitar el desarrollo de nuevas aplicaciones que tengan que tratar con datos de la universidad que ya se gestionen por nuestra aplicación y que requieran un sistema de permisos y de control de acceso. El mantenimiento

de los datos básicos de puede realizar desde nuestra herramienta de mantenimiento de datos y mediante las peticiones de la API se puede proveer de información útil a las aplicaciones conectadas y gestionar de forma eficiente los permisos.

Como se ha ido mencionando en los subapartados del apartado de desarrollo, los objetivos han sido logrados e implementados garantizando el cumplimiento de los requisitos iniciales.

Algo importante en la implementación de los requisitos, era la necesidad de hacerlo de forma atractiva para el usuario final, facilitando el uso de la aplicación y ahorrándole tiempo. Es por ello que se han implementado herramientas de multiselección, tablas dinámicas con sistemas de búsqueda y filtros de datos. También se ha conseguido minimizar el tiempo necesario de mantenimiento de la información mediante el proceso de importación automática de datos mediante archivos externos.

Otro aspecto muy importante era la seguridad. Para garantizar que el uso de la aplicación se hacía de forma controlada, se ha implementado una herramienta proporcionada por la universidad para integrar un sistema de acceso centralizado para todos los usuarios de la universidad, llamado SAC (Servei d'Autentificació Centralitzat) [13].

La planificación realizada y la metodología de desarrollo han resultado eficientes para el desarrollo del proyecto, aunque se subestimaron algunos aspectos como las horas necesarias para programar el gestor, donde se han tenido que invertir más horas de las previstas.

La integración del proyecto con el servidor final de la universidad no se tuvo en cuenta en la planificación y, finalmente, ha conllevado algunos problemas relacionados con las versiones de PHP utilizadas por parte del servidor final y el entorno local de desarrollo, siendo la del servidor un poco más antigua y no compatible.

Con todo esto, se espera que sea una aplicación de utilidad para la universidad y que facilite la gestión y mantenimiento de las encuestas y de cualquier otra aplicación que pueda encajar con lo que este proyecto intenta ofrecer.

Para finalizar con esta sección, dar las gracias a mi tutor de trabajo de fin de grado Jordi Pons, quien me ha guiado paso a paso en cómo desarrollar y mejorar todas las tareas que he ido realizando. Sin duda ha sido una buena experiencia y he aprendido mucho.

Referencias

[1] Universitat Autònoma de Barcelona, «UAB - Enquestes,» 23 Enero 2020. [En línea]. Available: https://www.uab.cat/enquestes/.

[2] Wikipedia, «Desarrollo en cascada,» [En línea]. Available: https://es.wikipedia.org/DesarrolloCascada. [Último acceso: 10 10 2020].

[3] Wikipedia, «API REST,» 03 Febrero 2021. [En línea]. Available: https://es.wikipedia.org/Transferencia[Último acceso: 03 Febrero 2021].

[4] Laravel, «Laravel,» 11 Diciembre 2020. [En línea]. Available: https://laravel.com/. [Último acceso: 11 Diciembre 2020].

[5] Bootstrap, «Bootstrap,» 28 Enero 2021. [En línea]. Available: https://getbootstrap.com/. [Último acceso: 28 Enero 2021].

[6] PhpSpreadsheet, «PhpSpreadsheet,» [En línea]. Available: https://phpspreadsheet.readthedocs.io/en/latest/. [Último acceso: 08 Noviembre 2020].

[7] Apache Friends, «XAMPP,» 27 Enero 2021. [En línea]. Available: https://www.apachefriends.org/es/index.html. [Último acceso: 27 Enero 2021].

[8] D. Axmark, A. Larsson y M. Widenius, «Workbench - MySQL,» [En línea]. Available: https://www.mysql.com/products/workbench/. [Último acceso: 26 09 2020].

[9] Postman, «Postman,» 05 Enero 2021. [En línea]. Available: https://www.postman.com/. [Último acceso: 05 Enero 2021].

[10] Nat Friedman, «GitHub,» [En línea]. Available: https://github.com/. [Último acceso: 01 Octubre 2020].

[11] Wikipedia, «Modelo-Vista-Controlador,». Available: https://es.wikipedia.org/wiki/Modelo. [Último acceso: 01 Octubre 2020].

[12] DataTables, «DataTables,» 04 Enero 2021. [En línea]. Available: https://www.datatables.net/. [Último acceso: 04 Enero 2021].

[13] UAB - Barcelona, «SAC – Central Authentication Service - UAB,» 05 Enero 2021. [En línea]. Available: https://si-peticions.uab.cat/. [Último acceso: 05 Enero 2021].

[14] B. Cook y C. Messina, «IETF Tools,» 06 Enero 2021. [En línea]. Available: https://tools.ietf.org/html/rfc6750. [Último acceso: 06 Enero 2021].

[15] D. Crockford, «JSON,» 1999. [En línea]. Available: https://www.json.org/json-en.html. [Último acceso: 26 09 2020].

[16] Wikipedia, «Arquitectura de microservicios,» [En línea]. Available: https://es.wikipedia.org/wiki/microservicios. [Último acceso: 05 Febrero 2021].

APÉNDICE

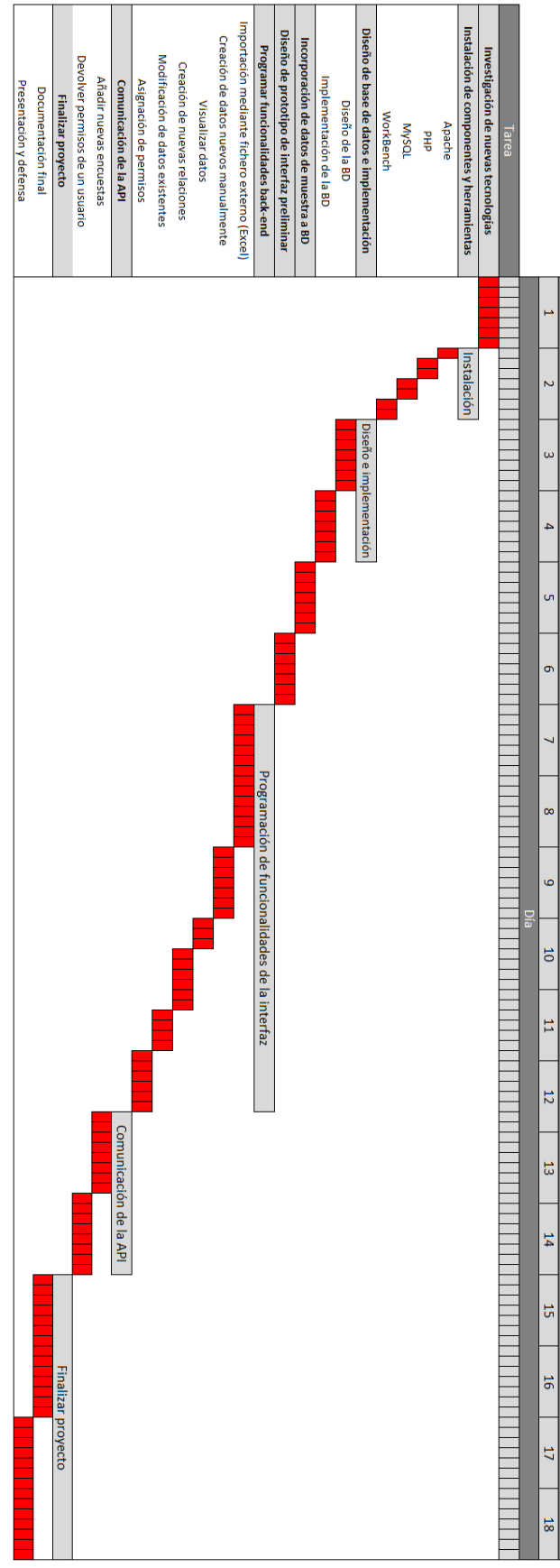

Ilustración 16. Planificación inicial del proyecto

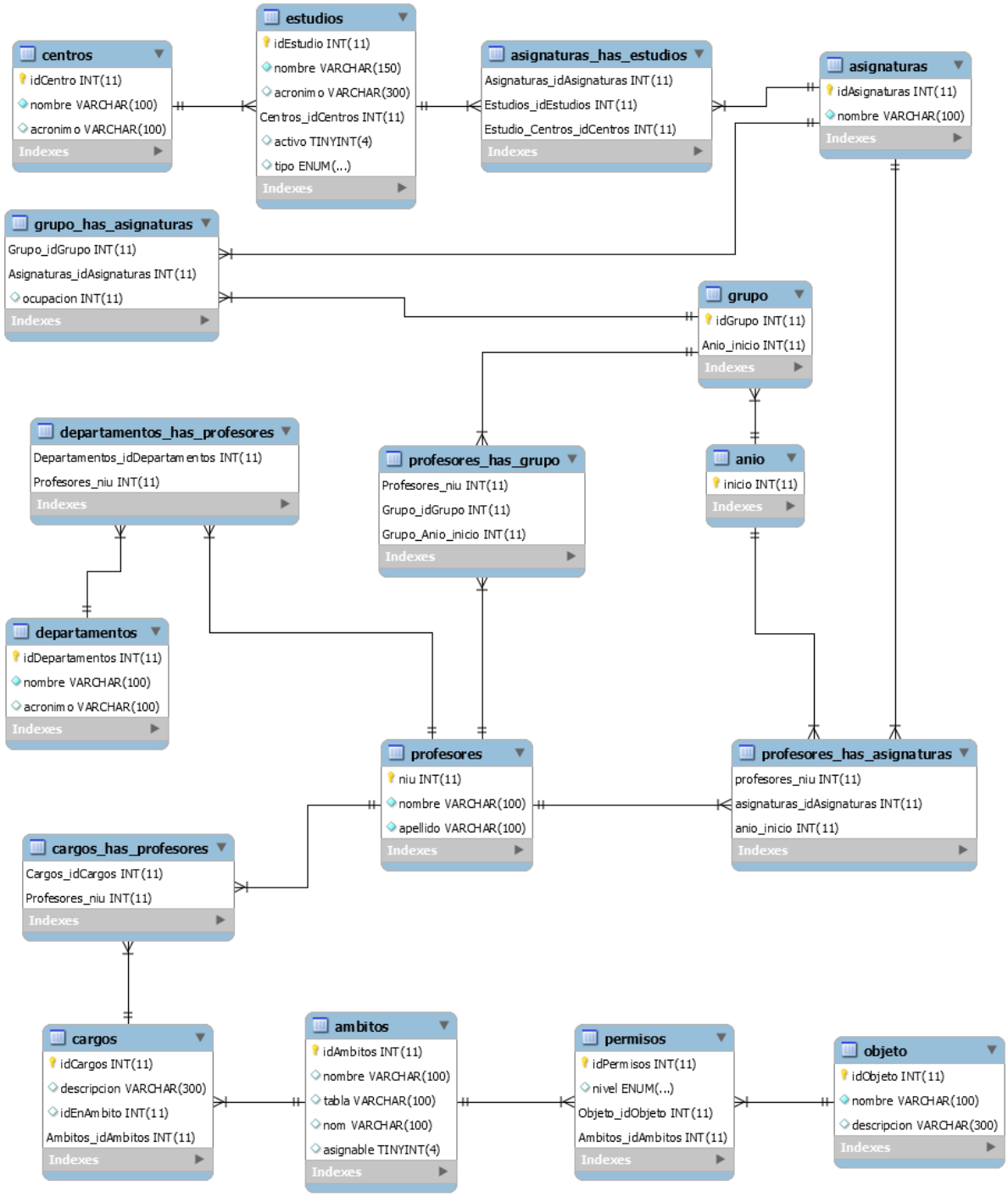

Ilustración 17. Base de datos

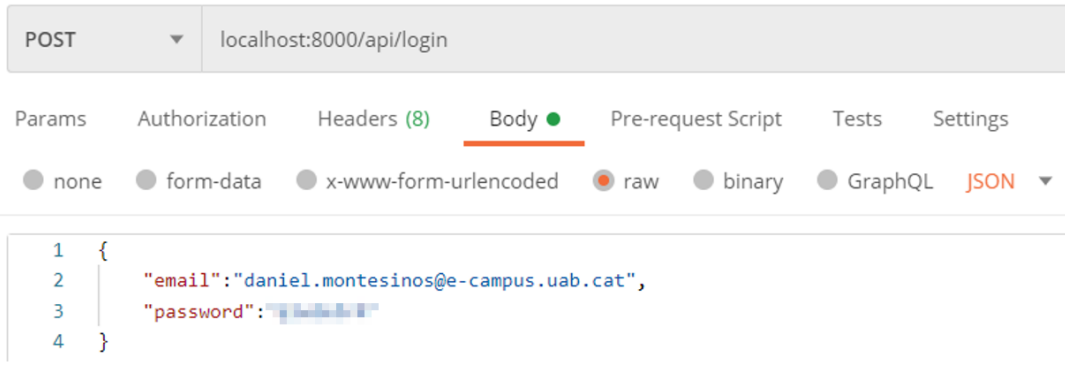

Ilustración 18. Petición de autenticación con Postman

Ilustración 19. Resultado de la autenticación con la API

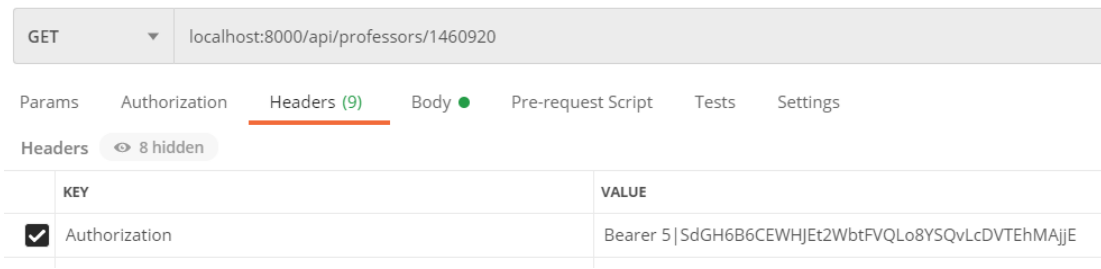

Ilustración 20. Ejemplo de petición GET - Profesor

Ilustración 21. Resultado de la petición

	Descripción de la petición	Ruta y formato
GET	Búsqueda de un profesor por NIU	/api/professors/{niu}
	Búsqueda de asignaturas, grupos y ocupación por curso de un profesor	/api/professors/assignatures/{niu}
	Búsqueda de departamentos de un profesor	/api/professors/departaments/{niu}
	Búsqueda de cargos de un profesor	/api/professors/carrecs/{niu}
	Búsqueda de objetos por ID	/api/objectes/cercar/id/{id}
	Búsqueda de objetos por nombre	/api/objectes/cercar/nom/{String} (%20% para añadir espacios)
	Búsqueda de objetos por descripción	/api/objectes/cercar/descripcio/{String}
	Retorno de permisos en función del ID del objeto	/api/objectes/permisos/{id}

POST	Añadir nuevo objeto a la DB	/api/objectes/afegir
		{
		"nom":"",
		"descripcio":""
		}

	Realizar autenticación de usuario	/api/login
		{
		"email":"daniel@e-campus.uab.cat",
		"password":"q1w2e3r4"
		}

DELETE	Eliminar objetos por ID	/api/objectes/eliminar/{id}

Ilustración 22. Lista de peticiones a la API